**校园社区平台**

作者姓名：赵安玉

专业名称：软件工程

指导教师：张玺 助教

**成都理工大学工程技术学院**

**学位论文诚信承诺书**

**本人慎重承诺和声明：**

1.本人已认真学习《学位论文作假行为处理办法》（中华人民共和国教育部第34号令）、《成都理工大学工程技术学院学位论文作假行为处理实施细则（试行）》（成理工教发〔2013〕30号）文件并已知晓教育部、学院对论文作假行为处理的有关规定，知晓论文作假可能导致作假者被取消学位申请资格、注销学位证书、开除学籍甚至被追究法律责任等后果。

2.本人已认真学习《成都理工大学工程技术学院毕业设计指导手册》，已知晓学院对论文撰写的内容和格式要求。

3.本人所提交的学位论文（题目：网上拍卖系统），是在指导教师指导下独立完成，本人对该论文的真实性、原创性负责。若论文按有关程序调查后被认定存在作假行为，本人自行承担相应的后果。

承诺人（学生签名）：

20 年 月 日

注：学位论文指向我校申请学士学位所提交的本科学生毕

**摘要**

随着信息技术的发展，以及移动终端设备普及，互联网完全融入了我们日常生活中。信息交流与共享无时无刻不出现在我们身边，网络上因此发布了各种社交软件与社交平台。但这些平台很少有专注面向高校，而高校又具有大量的用户群体和用户需求，所以校园社区平台具有研究和开发的价值。它可以满足高校内各个用户进行信息交流与共享的需求，无论是高校学生、老师还是普通职工，只要是高校认证用户都可以通过平台去发布或搜寻有效的信息。

校园社区平台，在实际生活主要用于老师、学生或者职工等高校自己内部群体发布或者搜寻信息。信息可以由文本、图片等多种类型描述，这样可以充分保证内容的多样性，比如高校新闻、食堂或宿舍的通知和寻物启事等等。但是，用户发布内容需要遵循平台所制定的规则，例如，平台对非法或粗俗敏感字进行过滤处理。平台管理人员由高校自己分配，他们对平台信息进行审核，这样有利于控制大部分信息的真实性。除此之外，平台用户也是高校自己对其进行身份认证授权，因此在一定程度上限制了用户群体，有保证了信息来源。

本平台设计实现技术采用SpringBoot作为后端的主体框架，整合MyBtis-Plus作为持久层框架连接MySQL数据库。前端选用Vue作为前端框架，页面的实现使用了Element-UI搭建。前后端数据交互方式则通过GraphQL实现。最后对平台各个功能模块进行测试，根据测试证实了各个模块所实现的功能基本满足了平台的日常使用，整体设计达到预期目标。

**关键词**：校园社区平台，SpringBoot，GraphQL，Redis，Vue

**Abstract**

With the development of information technology and the popularization of mobile terminal equipment, the Internet has been fully integrated into our daily life. Information exchange and sharing are all around us all the time, so all kinds of social software and social platform have been released on the network. However, few of these platforms focus on colleges and universities, which have a large number of user groups and user needs, so the campus community platform has the value of research and development. It can satisfy the demand of the users for information exchange and sharing, whether it is college students, the teacher or a worker, as long as the authenticated user can through the platform to publish or search for effective information.

Campus Community Platform, in real life, is mainly used for publishing or searching information among the internal groups of colleges and universities, such as teachers, students or staff. Information can be described by text, pictures and other types, which can fully ensure the diversity of content, such as college news, canteen or dormitory notices and missing notices, etc. However, users are required to follow the rules set by the platform, such as filtering illegal or vulgar and sensitive words. The platform managers are assigned by the universities themselves, and they check the information on the platform, which helps control the authenticity of most of the information. In addition, the platform users are also authorized by the universities themselves, so the user group is limited to a certain extent and the source of information is guaranteed.

The design and implementation technology of this platform adopts SpringBoot as the main framework of the back end and integrates MyBtis-Plus as the persistence layer framework to connect MySQL database. Vue is selected as the front-end framework, and Element-UI is used to build the page. The front and back end data interaction mode is realized by GraphQL. Finally, each functional module of the platform was tested. According to the test, it was confirmed that the functions realized by each module basically met the daily use of the platform, and the overall design reached the expected goal.

**Keywords**: Campus Community Platform，SpringBoot, GraphQL, Redis, Vue

**目录**

目录

[摘要 I](#_Toc672190820)

[Abstract III](#_Toc692699517)

[目录 V](#_Toc33027272)

[前言 - 1 -](#_Toc804616584)

[1 绪论 - 2 -](#_Toc616859315)

[1.1 项目开发背景与存在问题 - 2 -](#_Toc2105421661)

[1.2 主要工作和贡献 - 2 -](#_Toc1248602509)

[2 校园社区平台技术概述 - 4 -](#_Toc16984489)

[2.1 主要技术概念 - 4 -](#_Toc1591961026)

[3 平台架构设计 - 6 -](#_Toc1333538424)

[3.1 开发技术与开发环境 - 6 -](#_Toc513662309)

[3.2 平台架构 - 6 -](#_Toc1001802884)

[3.2.1 平台后端概述 - 7 -](#_Toc1915979416)

[3.2.2 平台前端概述 - 8 -](#_Toc1969112951)

[3.2.3 平台前后端数据交互概述 - 8 -](#_Toc250773604)

[4 平台数据库设计 - 9 -](#_Toc27911862)

[4.1 平台数据存储设计分析 - 9 -](#_Toc517008471)

[4.2 MySQL数据库表设计 - 9 -](#_Toc1596665281)

[4.3 Redis缓存设计 - 13 -](#_Toc1676105348)

[5 平台前后端设计 - 14 -](#_Toc1048353429)

[5.1 平台后端设计与实现 - 14 -](#_Toc1307634590)

[5.2 平台前端设计与实现 - 25 -](#_Toc280192363)

[5.3 平台前后数据交互策略及实现 - 26 -](#_Toc516737613)

[5.4 部分核心功能实现 - 29 -](#_Toc1055707026)

[总结 - 30 -](#_Toc1402284498)

[参考文献 - 31 -](#_Toc1168169070)

**前言**

随着互联网的不断发展，人们从网上获取信息的途径也多样化，但是所获取信息的问题也越来越多。对于大多数高校而言，无论学生还是在职人员，获取或发布信息多数借助于第三方平台。这样会带来的一系列问题，首先，平台上发布信息的真实性不能确定，例如，电信模仿诈骗历年来都会发生；其次，信息的时效性不能保证，不能保证信息及时和有效；然后，信息的可控性，学校或者个人难以通过第三方平台控制不实信息的传播。因此，推动高校自身的社区平台发展势在必行。

如今，Java作为生态最好和使用人数最多的开发语言，外加其开源社区Spring体系的发展，所以本平台开发选取它作为后端开发语言，并使用SpringBoot作为平后端主题框架并整合GraphQL、JWT和Mybatis-Plus等其他技术到项目中。前端目前Vue框架是国内使用最多与最流行开源JavaScript框架；因此，使用它配合前端UI框架Element-UI搭建前端应用。

现在无论是开发技术，还是开发工具互联网都有大量的资源，在本次毕设中选取对Java变成支持最好的开发工具IDEA 来进行开发。本次毕业设计参照了部分开源软件设计和现有网络上的视频教学，再结合大学所学习的专业知识，以及分析市面上社交软件功能，最后设计并开发了一款前后端分离的校园社区平台的Web应用。此外还使用了容器技术Docker搭建平台的MySQL数据库和Redis数据库服务；使用数据库管理工具Navicat、RDB，测试工具Postman、Chrome。对系统进行拓展和开发，来达到平台设计目标。

借助该平台，可以满足高校里各个群体发布或搜寻相关信息，而不用借助于其他平台。注册用户需要学校管理人员认证，以及发布信息也需要被核实，这样可以极大力度地控制信息的有效性和真实性。校内除了社团、学生会这样以人划分团体之外，食堂或宿舍这种以区域划分同样也可以作为一个团体使用官方账号来发布官方信息，这样极大的保证了平台用户体验友好性，

**1 绪论**

**1.1 项目开发背景与存在问题**

在互联网不断发展和信息技术的不断更新迭代现代生活中，每天在互联网上通信交流以及信息浏览的行为已经完全融入了人们的日常生活中。市面上的社交软件产品，大多数都为了容纳更广泛的用户群体以谋求利益而设计。对于高校学生、职员或团体等之间信息传递通常也采取QQ、微信和微博这类第三方平台。

高校内部借助于其他平台信息传递，所带来的问题也是显而易见，。首先，接收消息群体可能获取消息来源不统一，那么很难保证消息的及时性；其次，其他平台用户来源广泛，平台难以监管，用户容易被假冒，造成其他用户损失；除此之外，平台难以核实信息真实性并对其进行监管。通过本平台规范了消息来源，学校组织团体、个人等等只能通过学校平台管理方进行鉴定授权，从而限制用户范围；一些重要消息还需要进行审核，通过之后才能发布以保证信息的真实性。

**1.2 主要工作和贡献**

独立完成平台的需求分析，根据需求分分析再进行技术选型，技术选型从实现的合理性、难易程度、性能等等各方面进行综合对比，最终确定技术方案。根据需求完成数据库设计，然后便是后端的代码编写，在后端代码编写过程中发现有关于GraphQL的实际项目和教程，国内几乎没有，部分仅仅是介绍基本概念，不适用于实际项目。所以学习YouTube上教程以及阅读GraphQL Java Kickstart开源项目与其示例，除此之外，还浏览了大量有关GraphQL 帖子，最后形成一套较为完整的开发架构。完成后端代码编写测试之后，再进行前端的代码编写，同时进行前后端联调。最终完成整个项目，并编写本文。

# 2 校园社区平台技术概述

**2.1 主要技术概念**

1. 模块化设计概念和思想

模块化[程序设计](https://baike.baidu.com/item/%E7%A8%8B%E5%BA%8F%E8%AE%BE%E8%AE%A1/223952" \t "/home/anyu/Documents\\x/_blank)是指在[程序设计](https://baike.baidu.com/item/%E7%A8%8B%E5%BA%8F%E8%AE%BE%E8%AE%A1" \t "/home/anyu/Documents\x/_blank)时将一个大程序按照业务或者功能划分为若干小程序模块，每个模块完成与之对应的业务逻辑，这些模块之间建立合理的依赖联系，通过各个模块的互相协作完成整个系统设计的设计方法。利用模块化进行程序设计可以降低程序复杂度，同时使程序在设计、调试和维护等操作变的简单。

1. 前后端分离架构概念

前端一般泛指客户端和浏览器端，后端一般指服务端，前后端分离架构指在程序设计时，前端单独部署服务，其本身完成视图层渲染与用户交互，并通过请求获取所需要后端服务数据。后端接受请求，进行数据的逻辑处理，获取返回数据。前后端分离目的是关注度分离，后端只注重于数据和业务逻辑，而前端只注重视图和数据渲染。

1. Sping Boot 概念

Spring Boot是由Pivotal团队在Spring[框架](https://baike.baidu.com/item/%E6%A1%86%E6%9E%B6/1212667" \t "/home/anyu/Documents\\x/_blank)基础之上所设计的具有开箱机即用、快速构建Java应用优点的框架，它[简化](https://baike.baidu.com/item/%E7%AE%80%E5%8C%96/3374416" \t "/home/anyu/Documents\\x/_blank)了基于[Spring](https://baike.baidu.com/item/Spring/85061" \t "/home/anyu/Documents\\x/_blank)应用的初始环境搭建以及开发过程中大部分配置。该框架使用了约定的方式来进行自动配置，从而减少了开发人员在搭建或开发大部分配置。

1. Mybatis-Plus 概念

[MyBatis-Plus](https://github.com/baomidou/mybatis-plus" \t "/home/anyu/Documents\\x/_blank)是一个 [MyBatis](http://www.mybatis.org/mybatis-3/" \t "/home/anyu/Documents\\x/_blank)框架的增强工具，它在 MyBatis 的基础上只做增强不做改变，旨在为简化持久层开发、提高效率。它具有无侵入性、损耗小、支持Lambda形式调用、强大的CRUD操作、自定义全局通用操作等等优点。

1. GraphQL概念

GraphQL是一种可以用于 API 的查询语言也可以在运行时满足你数据查询的要求，它是Facebook公司于2012年在内部开发的数据查询语言，并在2015年对其开源，其作为替换RESTful架构体系的替代方案。 它对你 API 中的数据提供了一套易于理解的完整描述，使得客户端能够准确地获得它需要的数据，而且没有任何冗余，也让 API 更容易地随着时间推移而演进。

1. JWT (Json Web Token)概念

JWT是一个为了在网络应用环境间传递声明而设计基于JSON的开放标准(RFC 7519)，它被设计为紧凑、安全且自包含的方式，它的声明一般被用在身份提供者和服务提供者间传递被认证的用户身份信息。也可以增加一些额外的其它业务逻辑所必须的声明信息，该传递信息可以被验证和信任，因为它是经过数字签名。

# 3 平台架构设计

**3.1 开发技术与开发环境**

1. 开发环境
2. 操作系统：Linux 510
3. JDK版本：Java 11.0.7
4. 关系型数据库：MySQL 8.0.20
5. 缓存数据库：Redis
6. 开发工具
7. 后端开发开发工具：IDEA 2020.3
8. 前端项目开发工具：VSCode
9. 测试工具：Postman、Chrome
10. 开发技术
11. 容器技术：Docker
12. 后端框架：Spring Boot 2.3.4、MyBatis-Plus 3.4.0、JWT
13. 前端框架：Vue 2、Element-UI
14. 版本控制工具：Gradle 6.7.0

**3.2 平台架构**

平台是一个前后端分离的Web应用。后端使用了Gradle作为版本控制工具和模块管理工具；搭建基于Spring Boot 框架的Java Web应用架构; 整合Mybatis-Plus做持久层框架,对数据库进行操作。使用Vue框架搭建前端应用，利用Element-UI 进行快速页面搭建。使用GraphQL作为前后端通信方式。

### **3.2.1** 平台后端概述

各个模块遵循软件开发模块化设计原则，每个模块都只关注自己的功能或业务。模块架构依赖，如图3.1所示，其中ancf-msg模块、ancf-user模块、ancf-post模块等业务模块按MVC模式实现，它们是可以与前端进行交互的模块；理论上 ancf-auth模块、ancf-oss模块和ancf-cache模块等对外部模块只暴露业务接口，它们只负责对内部模块提供一系列服务。

1. ancf-webapp 模块，它是程序的主入口模块，最顶层模块；直接依赖于ancf-service-msg模块、ancf-service-user模块、ancf-service-post模块等业务模块。程序主配置文件存放该模块中。
2. ancf-service-msg 模块是校园社区平台用户间消息发送、平台消息推送等功能实现。
3. ancf-service-user 模块是校园社区平台与用户相关业务，包括用户注册、用户登录和头像修改等功能实现。
4. ancf-service-post 模块是校园社区论坛与帖子相关业务的实现，包括发帖，评论、点赞等功能实现。
5. ancf-common 模块是公共模块，最底层模块，基本上所有的模块都依赖于它。
6. ancf-service-auth 模块是平台权限认证、安全管理的实现。
7. ancf-service-oss模块是平台文件上传云端服务器的实现，借助于阿里云OSS服务。
8. ancf-sesrvice-cache 模块是系统缓存的业务实现。

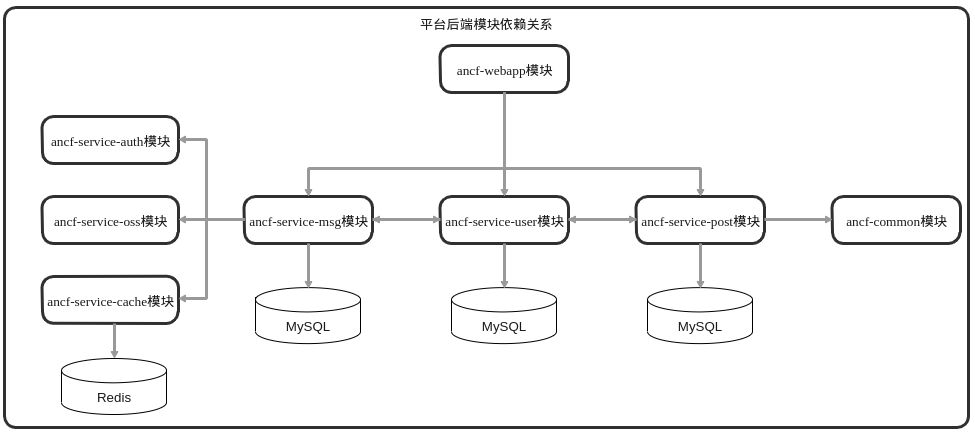


图3.1平台后端架构模块

### **3.2.2** 平台前端概述

前端是基于Vue框架构架的应用，使用Element-UI作为页面视图的快速构建。根据Vue数据双向绑定，从后端获取数据对页面进行实时动态渲染。

### **3.2.3** 平台前后端数据交互概述

使用GraphQL，后端使用GraphQL Java Kickstart 项目作为GraphQL服务端实现，该开源项目完整实现了GraphQL设计，对SpringBoot有良好的支持性和可扩展性。前端使用vue-apollo开源项目，通过vue插件的方式安装，搭建GraphQL客户端，配置服务地址，进行前后端数据交互。

# 4 平台数据库设计

本节主要介绍平台数据库、后端架构和前端架构的详细设计。

**4.1 平台数据存储设计分析**

（1）平台数据分析

对于平台数据分析，依据数据是否具有复用性将其分为两类，一类是有价值和可重复利用的数据，这类需要持久化存储，比如用户数据、帖子数据或者评论数据。另一类是具有时效性或没有持重复可用的价值的数据，比如Token和激活码这类信息，它们仅仅只需临时存储。除此之外，还有一部分读写热点的信息如果采用持久化方式性能会受到限。因此，除了持久化数据库之外，我们需要使用内存数据库。

1. 数据库类型分析

数据库按所储存数据类型可以划分为关系型数据库和NoSQL数据库两种类型。从存储结构分析，前者以二维表格预定义形式存储结构数据，具有良好的稳定性和可靠性；后者基于动态结构，将非结构化数据以块组合一起的形式存储，适用于数据和结构易于变化的数据存储。而本平台数据都具有明确的数据类型，因此，数据库选取关系型数据库。

对比多种数据库，关系型数据库使用MySQL，它作为平台主要的数据存储数据库；内存数据库使用Redis，它用来缓存部分实时热点数据或者具有时效性的数据。

**4.2 MySQL数据库表设计**

对于实体表，提供了通用的字段表示必要信息。例如create\_time字段表示创建时间、modified\_time字段表示修改时间、status字段表示数据状态信息。

1. 用户表 user

用户表的表结构信息如表4.1所示，它包含平台用户的个人信息、认证信息以及一些账号状态信息。账号、id和邮箱都是唯一的，可用作登录。头像存储的是阿里云对象存储服务的图片地址。

表4.1 user表逻辑结构

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 长度 | 是否 NULL | 注释 |
| id | int | 10 | NOT NULL | ID |
| account | char | 16 | NOT NULL | 账号 |
| gender | tinyint | 1 | NOT NULL | 性别 |
| age | tinyint | 3 | NOT NULL | 年龄 |
| birthday | datetime | - | NOT NULL | 出生日期 |
| nickname | char | 9 | NOT NULL | 昵称 |
| real\_name | char | 6 | NULL | 真实姓名 |
| avatar | char | 100 | NOT NULL | 头像存储地址 |
| email | char | 25 | NULL | 邮箱地址 |
| mobile | char | 11 | NULL | 手机号 |
| salt | char | 5 | NOT NULL | 密码加盐 |
| password | char | 50 | NOT NULL | 密码 |
| activation | tinyint | 1 | NOT NULL | 账号是否激活 |
| status | tinyint | 1 | NOT NULL | 用户状态 |
| create\_time | datetime | - | NOT NULL | 用户创建时间 |
| modified\_time | datetime | - | NOT NULL | 用户修改时间 |

1. 评论表 comment

评论表的表结构信息如表4.2所示，用于帖子的评论、评论的评论、或者评论的回复的表示，帖子的评论的字段entity\_type 必须为 1,target\_id 必须为null；评论的评论的字段entity\_type 必须为2，target\_id为null，评论的回复entity\_type 为3，target\_id 为null。

表 4.2 comment表逻辑结构

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 长度 | 是否 NULL | 注释 |
| id | int | 10 | NOT NULL | 评论id |
| content | varchar | 255 | NOT NULL | 内容 |
| user\_id | int | 10 | NOT NULL | 评论者id |
| entity\_id | int | 10 | NOT NULL | 评论的实体id |
| target\_id | int | 10 | NULL | 被评论者的id |
| entity\_type | tinyint | 1 | NOT NULL | 评论实体类型 |
| status | tinyint | 1 | NOT NULL | 评论状态 |
| create\_time | datetime | - | NOT NULL | 评论创建时间 |
| modifid\_time | datetime | - | NOT NULL | 评论修改时间 |

1. 帖子表 post

帖子表的表结构信息如表4.3 所示，他是用于存储用户平台帖子信息，其信息包含了帖子状态，发布者的id，帖子标题，帖子类型和标题等必要信息。

表4.3 post表逻辑结构

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 长度 | 是否 NULL | 注释 |
| id | int | 10 | NOT NULL | 帖子id |
| type | tinyint | 1 | NOT NULL | 帖子类型 |
| user\_id | int | 10 | NOT NULL | 发帖者id |
| title | varchar | 20 | NOT NULL | 帖子标题 |
| content | text | - | NOT NULL | 内容 |
| cmt\_num | int | 10 | NOT NULL | 评论数 |
| score | float | - | NOT NULL | 得分 |
| status | tinyint | 1 | NOT NULL | 帖子状态 |
| create\_time | datetime | - | NOT NULL | 创建时间 |
| modified\_time | datetime | - | NOT NULL | 修改时间 |

1. 消息表 message

消息表的表结构信息如表4.4所示，平台中每一个用户都可以相互已私信的方式进行交流。两个人之间的私信具有相同chart\_id，chart\_id由from\_id和to\_id根据特定规则生成，交流的两个用户的信息通过chart\_id来在数据库快速查询获取。在通过其他字段来判别信息由谁发送与由谁接受。

表4.4 message表逻辑结构

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 类型 | 长度 | 是否 NULL | 注释 |
| id | int | 10 | NOT NULL | 消息id |
| from\_id | int | 10 | NOT NULL | 发送者id |
| to\_id | int | 10 | NOT NULL | 接受者id |
| chart\_id | varchar | 20 | NOT NULL | 聊天id |
| content | varchar | 255 | NOT NULL | 内容 |
| status | tinyint | 1 | NOT NULL | 状态 |
| create\_time | datetime | - | NOT NULL | 创建时间 |
| modified\_time | datetime | - | NOT NULL | 修改时间 |

**4.3 Redis缓存设计**

1. 设计缘由以及问题分析

首先，考虑到部分数据修改和查询频繁，直接访问MySQL数据库会造成其压力过大，从而导致程序效率问题。其次，像激活码这种临时的一次性数据，这类数据具有时效性，过期的数据没有任何可利用价值。如果存放到MySQL数据库中，仅仅只会被使用一次而且没有重复利用的价值，显然不合理。因此，我为平台设计数据缓存服务。

1. 设计方案与实现

借鉴目前缓存设计的方案，我在本地缓存、Redis以及Memcached三者之间进行对比选择。首先，三者都是以Key-Value的形式存储，本地缓存适用于数据量较少且对数据操纵简单的数据，Redis可以支持更复杂的操作，如设置过期时间；所以本地缓存方案被否定。然后，Redis支持String、Hash、Set、List和Sorted Set等五种数据操作，而Memcached只支持String，并且Redis支持数据持久化，而Memcached不支持。综上所述，从技术和实现功能难易程度两个方面考虑，选择Redis作为平台的缓存数据库是最合理方案。

# 5 平台前后端设计

**5.1 平台后端设计与实现**

本章节介绍后端平台架构，各个模块实现的功能和所使用技术。以及它们在平台具有的作用。

**5.1.1 模块功能设计实现**

1. ancf-common模块

本模块是平台的公共模块，将其他模块常用并且不耦合的功能单独抽离封装成工具类或者组件，共所有模块使用，提高代码复用率。

1. CommonUtils工具类

本工具类提供给其他模块的公用工具类，里面包括了BASE64加密解密、生成随机字符串、生成随机数字字符串和MD5加密等静态方法。

1. MailClient 系统邮件发送组件类

邮件客户端使用了springboot自带的邮件依赖，对其进行了自定义封装，将其实例注入到IOC容器供其他模块使用，通过调用其实例sendMail方法，传入一个或多个收件人id、发件人id和邮件内容发送邮件。

1. SensitiveFilter 敏感字文本过滤类

对于一个公共的平台，需要对敏感信息进行过滤处理，SensitiveFilter 正是平台的敏感词过滤处理工具类。其主要实现借助于字典树这一数据结构。其类图如图5.1所示，init方法在初始化时会读取平台自定义敏感字源文件，遍历其中每一个词组，并将其通过saveKeywordToRoot 方法加入到敏感字字典树中。最后生成一个包含有所有敏感字信息的字典树。filter方法对传入的文本文件借助于已生成的敏感字字典树进行算法处理，最后返回过滤的文本。

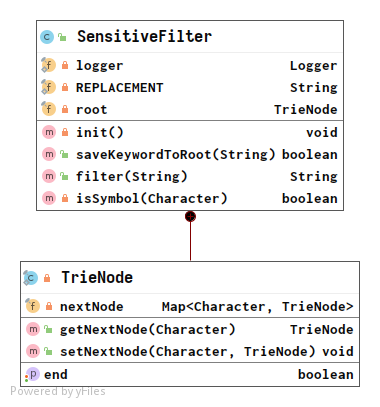
****

图5.1 SensitiveFilter类图

1. 数据库字典枚举类实现

数据库的某一些字段表示的值或者内容范围较小且固定，在设计数据库这些字段时，考虑到性能与合理性，通常字段类型使用tinyint类型，与Java中的Integer相互对应；除此之外，考虑到数字所表示的信息在实际代码开发中对开发人员不够清晰，因此使用枚举来定义这些字段。本平台的数据库字典枚举都继承IEmnuCode接口，通过实现该接口用来表示这是数据库字典，其存储是自动映射使用MyBatis-Plus实现。类图如图5.2所示，getCode 方法方法获取该枚举所存入数据库的值，getDesc 方法获取该枚举的描述信息。

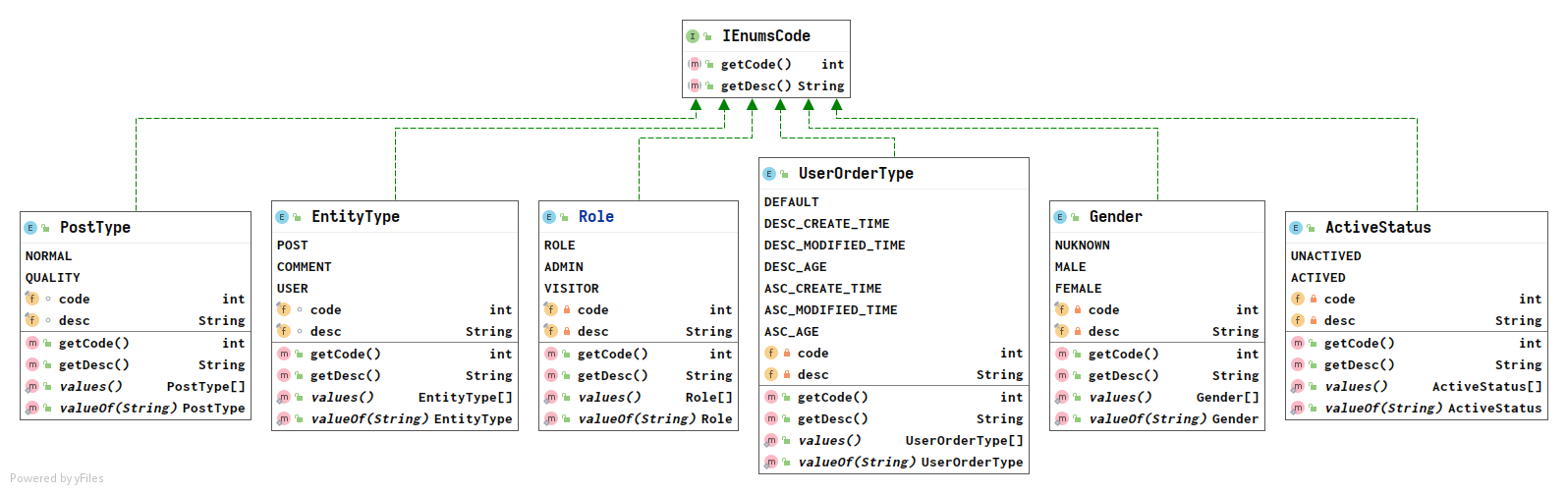
****

图 5.2数据库字典实现类图

1. 业务处理结果状态枚举类

考虑到任何有意义的请求，服务器端都会进行业务逻辑操作，并最终得到有且只有一个结果状态，将其返回给客户端。因此，平台在设计时针对此类结果状态进行统一化和规范化。所有业务处理结果的状态的枚举如图5.3所示，统一实现IResultType接口，该接口有且只有三个方法，分别是isSuccess方法、 getCode方法和getMessage方法。isSuccess方法表示业务处理结果是否成功，其返回类型为布尔类型；getCode方法返回业务处理结果的业务代码，其返回类型为Integer类型；getMessage方法返回业务处理结果的业务结果信息，其返回类型为String类型。

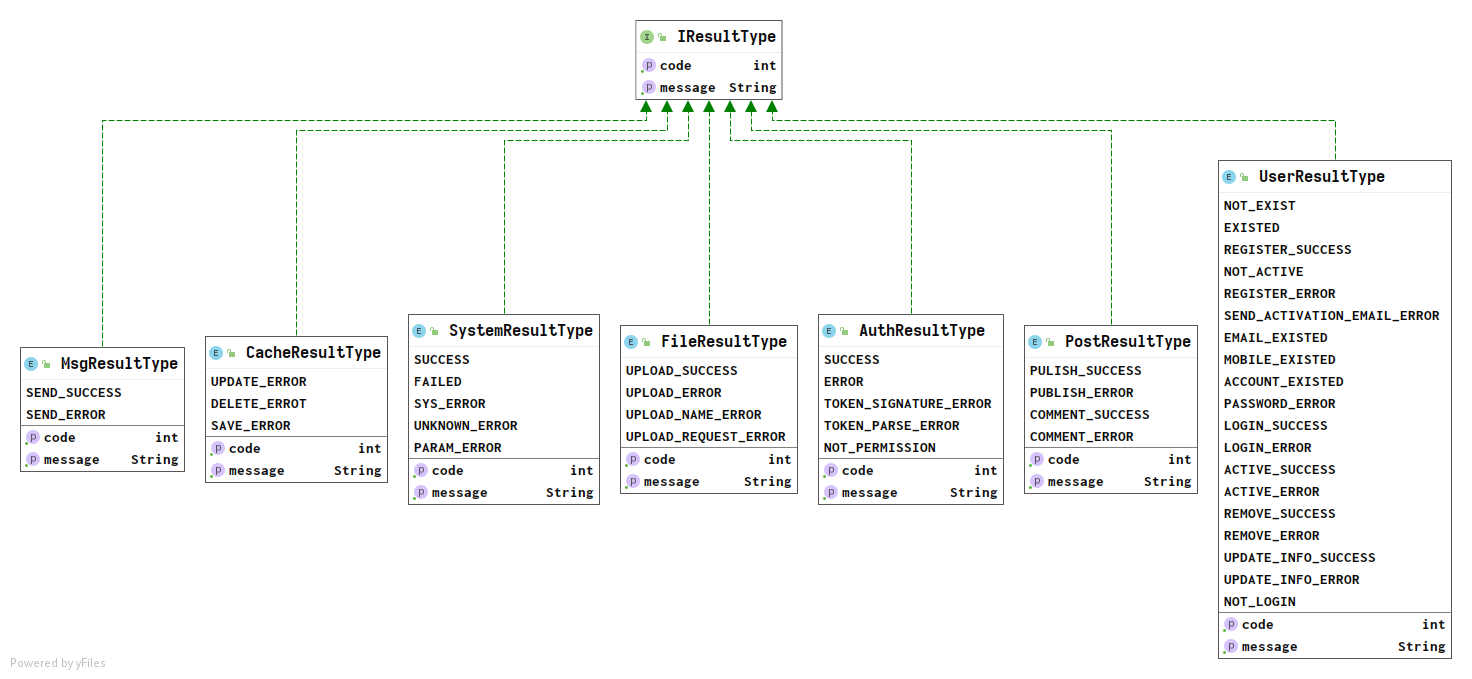
****

图5.3 IResult类图

1. 自定义全局异常类

本平台所自定义的异常类继承于RunTimeException异常类，添加了code属性，重写了getMessage方法，用于返回业务结果状态信息。提供了一个静态 causeBy方法，用于通过业务处理结果状态，生成异常信息。

1. 全局异常处理

全局异常处理类实现GraphQLErrorHandler接口，重写了processErrors方法。只返回给客户端业务处理错误信息与其业务代码。

1. 统一结果返回

其用来描述Mutation类型操作结果的数据，并将其返回给客户端。其类图如图4.8所示，提供了静态的with方法，可通过传入业务处理结果状态枚举类，来生成统一结果。

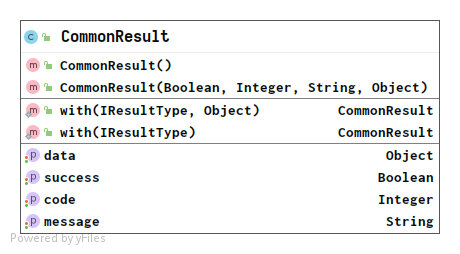


图5.4CommonResult类图

1. ancf-service-cache模块

本模块构建了平台缓存服务，其包含了平台所有与缓存有关的实现与配置，借助于Spring 实现的RedisTemplate直接对Redis数据库进行操作。

1. Redis 键生成工具类

缓存信息时，需要根据某些信息生成特定的键，方便再次从缓存中获取缓存信息，因此设计了该工具类。

1. 缓存服务接口及实现

其实现类图如图5.5所示，其他模块使用缓存是只能通过调用缓存接口来使用缓存服务。目前平台针对账号激活的激活码、帖子点赞数和点赞状态进行缓存实现。

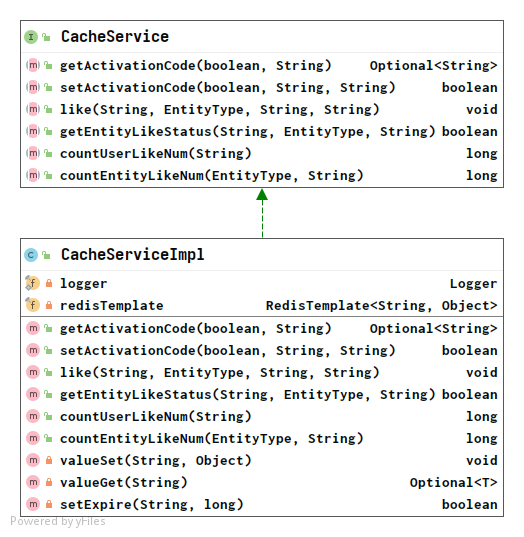
****

图5.5缓存服务实现类图

1. ancf-service-post 模块

本模块构建了所有帖子相关的服务功能，其依赖了ancf-service-user模块和ancf-service-cache模块。

1. 评论API数据交互层

它是帖子评论交互实现，其类图如图5.6所示，一个实现GraphQueryResolver接口，表明其是一个GraphQL的数据查询操作，另一个实现GraphMutationResolver接口，表明其是一个GraphQL的数据变更操作。功能包含评论、分页查询评论列表和获取回复。

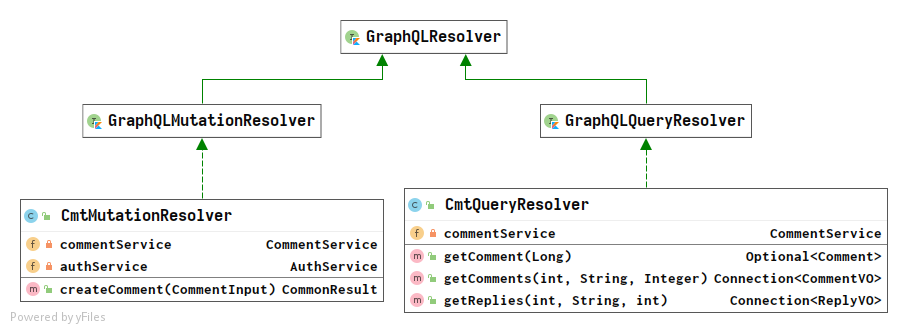
****

图5.6评论Resolver类图

1. 帖子API数据交互层

帖子数据交互接口类图如图5.7所示，同样分别实现两个不同Resolver接口，其功能包含发布帖子、分页查询帖子和根据id查询帖子详情。

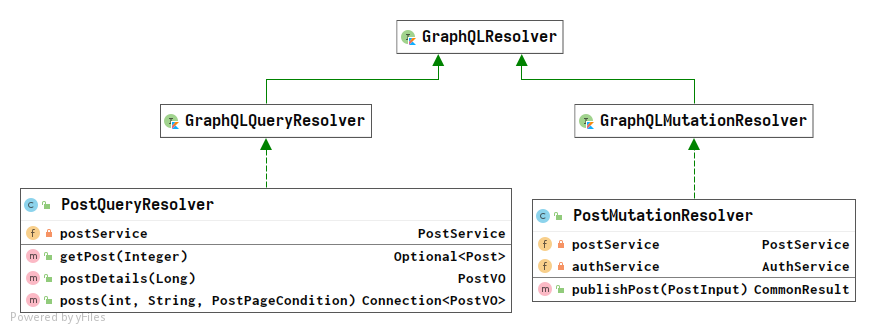
****

图5.7帖子Resolver类图

1. 帖子服务层

帖子服务接口和实现类类图如图4.12所示，是帖子业务逻辑的具体实现，包含根据帖子id查询、分页查询帖子列表、发布帖子和帖子实体类类型转换等具体功能实现。

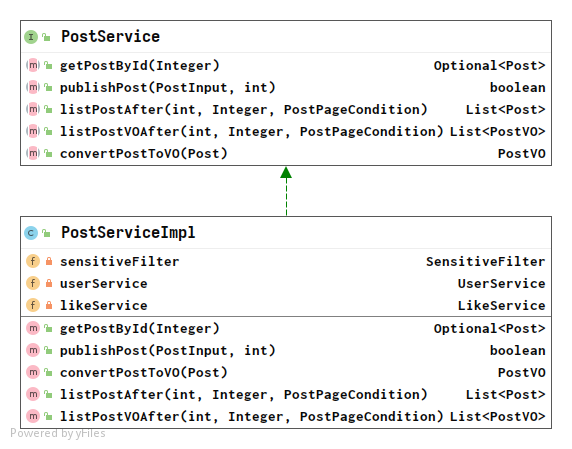
****

图5.8帖子服务层类图

1. 点赞业务层

帖子和评论点赞业务的接口与逻辑实现，其类图如图5.9所示，其依赖于缓存服务，有查询评论的点赞数、获取点赞状态、点赞以及取消点赞等功能具体实现。

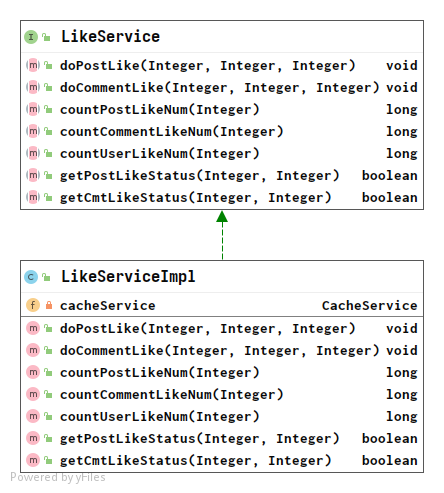


图5.9点赞服务实现类类图

1. 评论业务层实现

评论业务层接口及实现类，其类图如图5.10所示，实现根据不同的实体查询评论、创建评论、实体模型转换以及分页查询评论列表等功能。

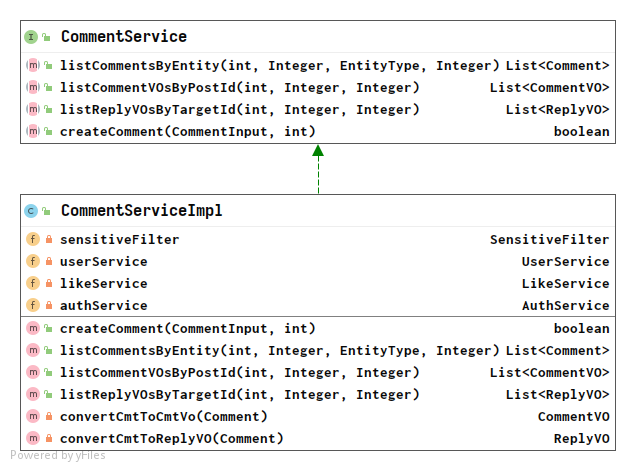
****

图5.10评论服务类图实现

1. ancf-service-oss模块

考虑到图片资源这类资源可以存储到远程服务器，从而降低资源请求对本地服务器的压力，并且其访问速度够快。所以该模块借助于阿里云对象存储服务，实现将图片资源存储到阿里云服务器。实现头像的存储和背景图片存储功能。

1. ancf-service-auth模块

（1）认证请求上下文构建

平台的安全认证实现，重写GraphQL的请求上下文，其类图如图5.11所示，在其中加入JWT解析，注入认证对象。当请求进入时，AncfGqlContextBuilder会调用认证服务通过Token去解析和验证登录用户信息，如果成功会将解析的已登录用户信息注入DataLoaderRegistry同时存储到本模块使用ThreadLocal实现的缓存中，供其他模块使用。然后初始化AncfGqlSocketContext或者AncfGqlHttpContext。

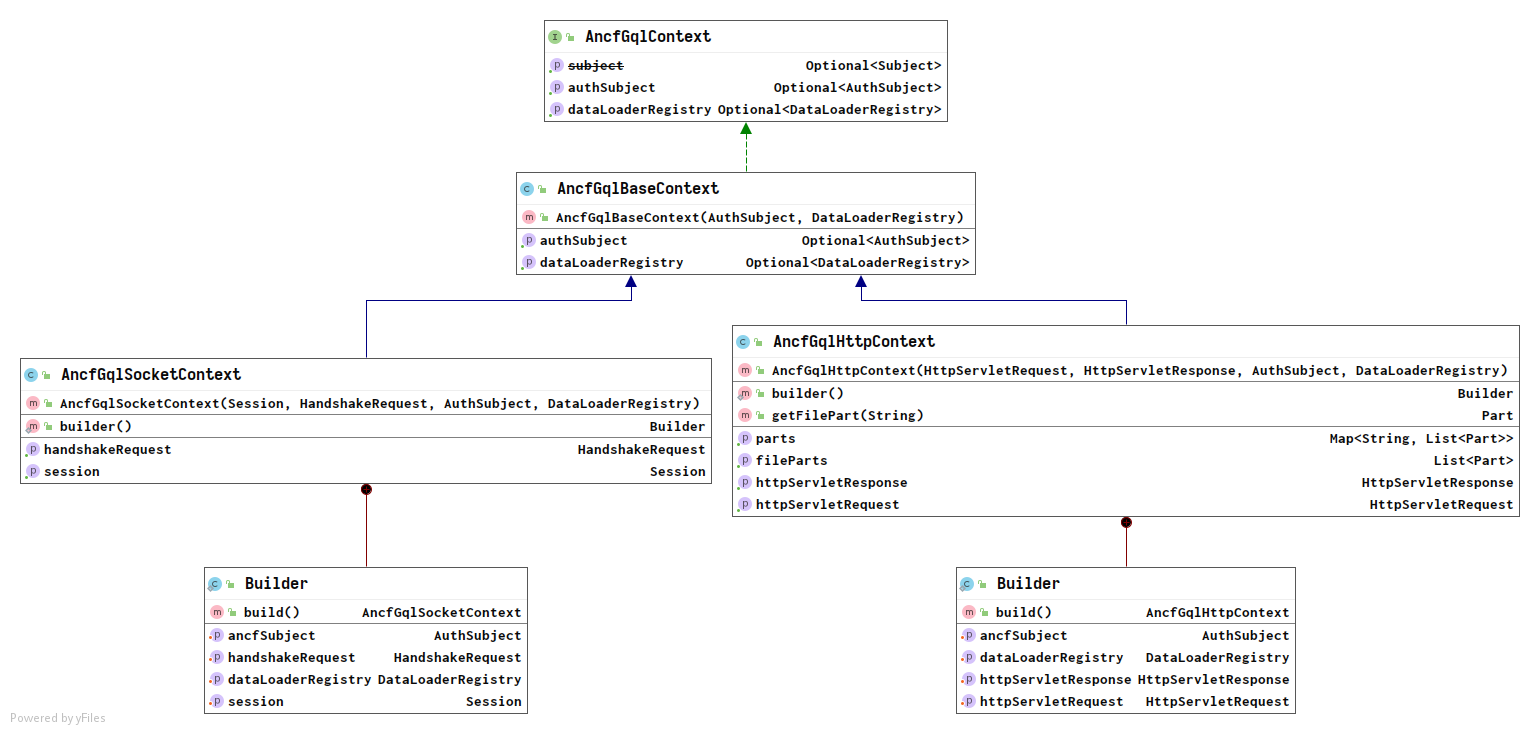
****

图 5.11GraphQL请求上下文构建实现类图

（2）JWT 服务

JWT服务类类图如图4.17所示，JwtProperties属性配置类[通过@EnableAutoConfigProperties注解，实现JwtProperties相关的属性与application.yaml配置文件绑定。JwtHelper实现一系列](mailto:通过@EnableAutoConfigProperties注解，实现JwtProperties相关的属性与application.yaml配置文件绑定。JwtHelper实现一系列jwt)JWT具体功能，比如生成、解析或验证Token、获取用户名和角色信息等等。

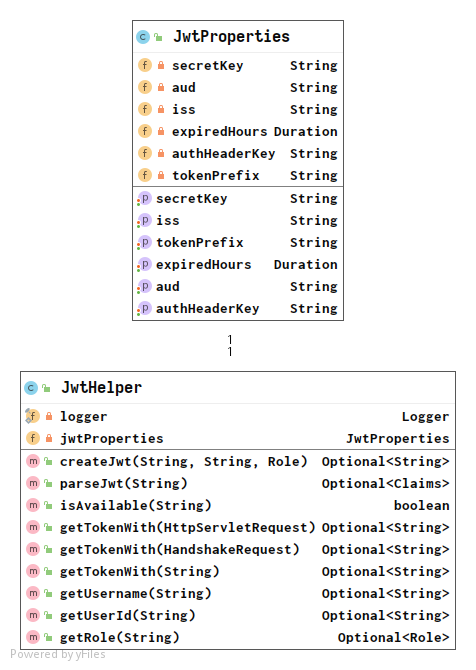
****

图4.17 JWT配置类与服务类类图

（3）认证授权服务

服务实现类图如图4.18所示，其提供了给外部模块访问的接口，创建JWT，解析JWT，获取当前认证用户信息，认证用户角色信息等业务实现。

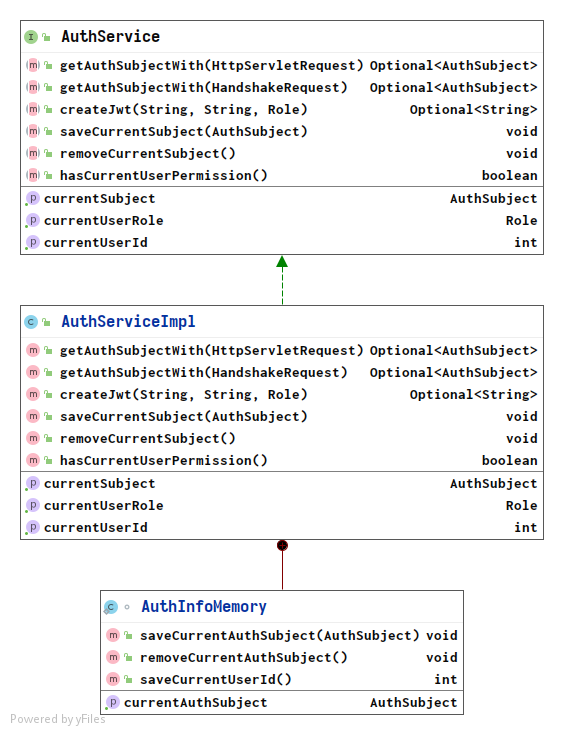
****

图4.18认证授权业务实现服务类类图

1. ancf-service-user模块

（1）用户API数据交互层

与前端交互的接口实现类图如图4.18所示，提供登录、注册、查询用户，用户信息查询，用户注册，用户登录，用户密码修改，用户激活，用户注销，更新头像等等交互接口。

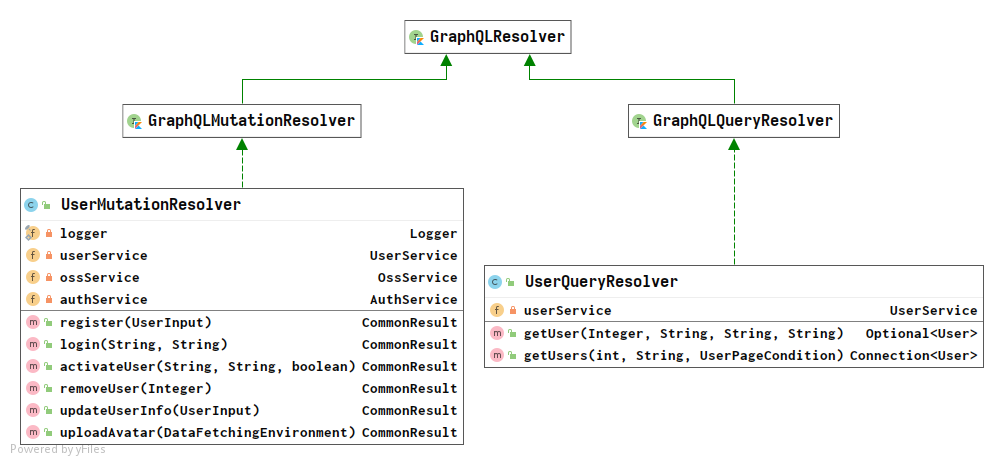
****

图4.19用户Resolver类图

（2）用户服务实现类

与用户相关的操作都由该服务完成，其类图如图4.19所示，用户信息查询，用户注册，用户登录，用户密码修改，用户激活，用户注销，更新头像等业务的具体实现。

1. ancf-service-msg模块

平台信息交流，用户私信交流等功实现。

1. ancf-webapp模块

程序顶层模块，包含程序的主启动类，程序的全局配置文件application.yaml也存放在当前模块,程序GraphQL基础模板文件，也存放在该模块。

**5.2 平台前端设计与实现**

本章节介绍前端的技术选型与设计

前端应用，目前有三大主流JavaScript框架，React，Angular和Vue,它们各自具有以下特点，React用于构建用户界面，具有声明式和组件化特点，在UI渲染过程中，通过在虚拟DOM中的微操作来实现对实际DOM的局部更新，渲染速度很快；Angular是一个应用设计框架与开发平台，用于创建高效、复杂、精致的单页面应用，它具有原生、渐进式应用与跨平台等特性；Vue是一个渐进式轻量级JavaScript框架，本身具有React与Angular大部分特性，支持数据双向绑定、指令、插件化和易用灵活高效特点，在国内有良好的生态。

React虽然速度快兼容性好，但是他是单向数据流，不支持数据双向绑定。Angular具有数据双向绑定、指令、服务和模板等比较完善的前端框架功能，但是学习成本高，文档例子少。而Vue具有它们的大部分优点，文档丰富，所以进行对比之后，选取Vue作为前端基础框架。

**5.3 平台前后数据交互策略及实现**

本节主要介绍前后端服务数据交互方案和实现。

5.3.1 交互技术选取技术分析

1. RESTful风格架构分析

常见的前后端交互方式，一般选用RESTful风格的API设计，但是它具有以下缺点。

首先，每创建一个交互接口，都会赋予一个URL和请求方式，当接口数达到一定规模，那么对于接口的维护便变的异常困难；其次，对于不同接口返回实体的某些字段并不需要，解决办法往往是创建与之对应的VO对象，但这种方式也过于繁琐和冗余；在很多实际业务中，API往往需要进行版本更新维护，比如废弃或者添加某个实体字段，那么就需要对接口的参数实体，返回实体进行维护。

它的优点架构设计简单，具有完整和成熟的技术体系，开发者都了解并使用过。

1. GraphQL架构分析

GraphQL作为一种可替代 RESTful风格的API查询语言，其本身具有以下优点。

首先，一个请求可以获取多个资源，查询时不仅能获取资源属性，还能通过资源间的关系进一步查询。即使是比较慢的移动网络连接下，使用 GraphQL 的应用也能表现得足够迅速。其次，它本身具有自己的类型，查询时通过类型和字段约束的方式进行结果组织，精确的查询结果展示和单一的URL接口，不仅节省了网络资源，而且避免了大量的URL接口，而非入口端点。因此具有通过一个单一入口端点获取到你所有的数据能力。除此之外， 使用GraphQL可以避免让你的整个应用被限制于特定存储引擎，它完全可以使你应用共享一套 API。能够更充分利用到你的现有数据和代码。你仅仅需要为类型系统的字段编写解析函数，GraphQL 就可以通过优化并发的方式来调用它们。

它的缺点，尽管Facebook、Twitter和Github等等大公司在使用，但是它们并未公开内部详细设计，如今一些社区组织正在不断完善其设计，整个技术生态并不成熟，整合到Spring Boot中，开发仍然略微复杂。

1. 技术选择

由于本平台是具有论坛系统特征，帖子属性可能来源多个存储引擎，比如缓存数据库，同时不同场景需要特定的属性，场景较为复杂，因此，本平台采用GraphGL架构风格。

5.3.2数据交互设计与实现

GraphQL Java语言的实现有几个开源项目，因为平台是基于springboot，需要有已整合置好starter，选择GraphQL Java Kickstart开源项目框架，作为后端GraphQL服务。

1. Query交互设计实现

平台的查询接口设计，所有查询交互模型如图4.1所示；其中分页查询的实现基于Relay的方式。

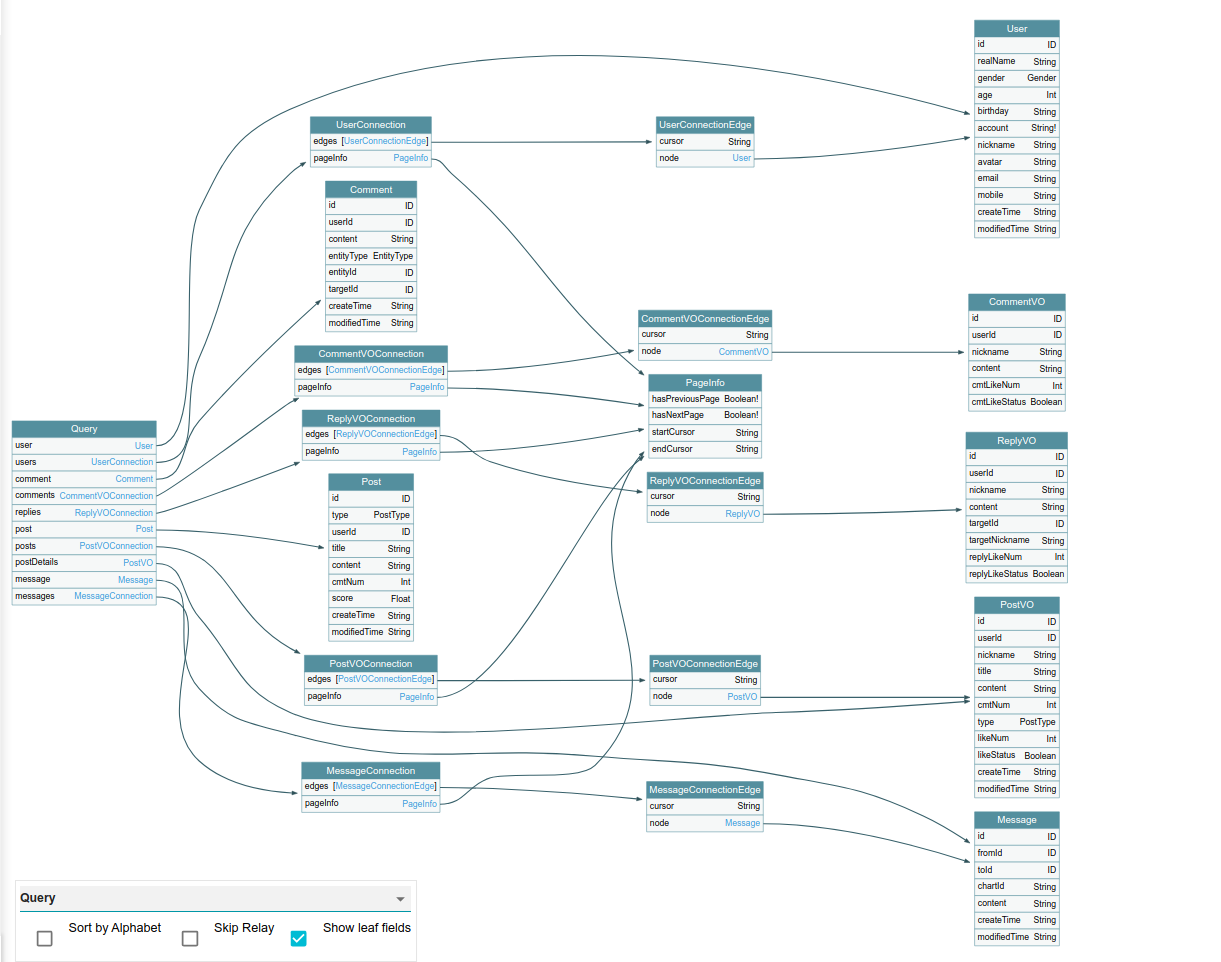


图4.1 Query交互结构模型

1. Mutation 交互设计及实现

平台的变更操作，平台所有变更操作交互模型如图4.2所示，变更操作接口返回类型，统一使用CommonResult。

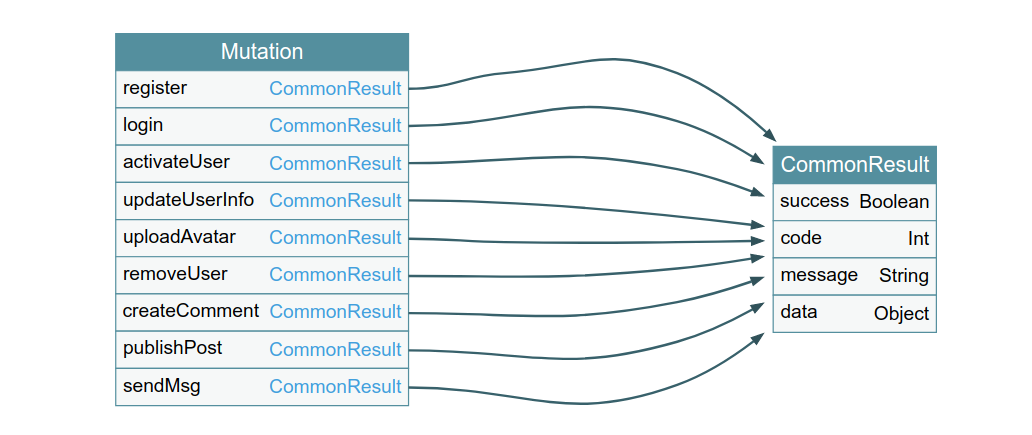


图4.2 Mutation交互结构模型

**5.4 部分核心功能实现**

1. 发布帖子流程

帖子发布需要用户登录，当用户在前端页面编辑好帖子内容时并点击发布，后端PostMutationResolver中publishPost方法会接受数据并将其反序列化成PostInput对象，此过程中会对帖子进行参数校验，如果校验失败返回字段错误信息给前端显示，成功该对象调用toEntity方法将根据其属性值实例化成存储数据库的PO对象。将该对象作为参数传入帖子服务层的publishPost中，在服务层方法中，首先，会通过sensitiveFilter的filter方法对帖子的标题和内容进行敏感字过滤处理并再通过HtmlUtils的htmlEscape方法对字符进行编码处理；其次，会通过authService中getCurrentUserId方法获取当前登录用户的id，并将其赋值给帖子的userId属性；然后再初始化帖子其他属性；最后将使用MyBtis-Plus中实现的save方法保存到数据库中，成功后将其成功信息返回给前端，前端弹出成功提示框。

1. 主页面帖子列表显示

当浏览器访问主页面时，客户端会像服务端发起获取最新帖子列表请求，

1. 帖子详情页

帖子详情页包含对帖子详情和其评论列表。

1. 个人信息修改

个人信息页，可以对个人信息的修改，如密码、头像。

**总结**

本文从零开始设计和实现校园社交平台，经过最初的可行性分析、需求分析和数据库设计，到中期开发、单元测试和前后端联调，以及最后的程序优化等一系列软件开发流程。亲身体验了软件工程中这些软件开发生命周期各个步骤，认识到软件工程这门学科对于软件开发的重要性和必要性。

在完成毕设这一段时间中，我将Java知识进行系统的查漏补缺学习了一遍，对于已掌握的知识，更加熟悉、更深层次了解其本质，对于以前疏漏的知识，学习起来更加容易并且印象更深刻。除此之外，还自学了GraphQL、JWT和Vue等等大量前沿和热门的开发知识并且将其使用到项目之中，尤其对于Spring Boot框架使用更加熟练并明白其大部分设计理念，同时学习到更多用于实际开发重要特性。

经过本次的毕业设计，深刻的理解到了算法的重要性；比如，在敏感字过滤设计实现方案中，从最开始使用二分法遍历，再到使用字典树这种数据结构，效率成倍变高，回想大学刚开始学习编码时，老师常叮嘱我们算法和数据结构是程序设计的灵魂，无疑是我现在对其认知。同时体验到了设计模式对于好的程序设计与程序开发不可或缺重要性。例如，在JWT运用项目时，未使用拦截器，采取直接重写原有的GrapQL请求上下文植入JWT解析，其中就使用到创建者模式和简单工厂模式，是程序逻辑更加清晰，实现更加简单。

本次毕业设计是对大学所学习的知识进行实践检验，从中意识到了很多自己不足地方，并正努力改正，这可能会深深地影响我以后的职业发展。

**参考文献**